OKTICKET S.L.

API - Funcionalidades extendidas y ejemplos

v1.3 - 20 de Mayo de 2020

1. OBJETIVO Y ALCANCE

El presente documento presenta diversos ejemplos y casos de uso avanzados de la API de Okticket, previamente definida en el documento Guía de Uso.

Los ejemplos están orientados a aquellos desarrolladores que ya hayan tenido una primera toma de contacto con la API y conozcan los detalles básicos de la misma: autenticación, paginación, etc.

2. EJEMPLOS

Se agrupan los ejemplos mostrados por bloque funcional o entidad.

2.1. Gestión de campos de tipo Lista externa

En Okticket existe la posibilidad de activar y configurar nuevos campos personalizables por cliente. Estos campos pueden activarse a nivel de Gasto u Hoja de gastos, y pueden ser distintos en función del tipo de gasto, la empresa, la categoría del gasto o el rol de usuario.

Dentro de los campos personalizable disponibles en Okticket, existen el tipo de campo **Lista Externa**, que permite que sus valores se obtengan de una tabla alimentada vía API.

El campo aparece en la APP y el gestor de Okticket como un selector (que puede ser múltiple), que muestra una lista de textos (etiqueta) y que guarda un valor al ser seleccionado, en el gasto u hoja correspondiente. Este valor puede luego recuperarse dentro de la entidad correspondiente, en el atributo `custom_fields.<id_campo>`.

Cada elemento de la lista se compondrá de cuatro atributos principales:

- `label`: Etiqueta, lo que se muestra a los usuarios en pantalla
- `value`: Valor, lo que se guardar en el modelo
- `global`: Si es global, el elemento le aparecerá a todos los usuarios; si no, solo a los que se configuren para ello.
- `active`: Si está activo, se mostrará en la lista.

Se muestran a continuación los ejemplos de llamadas necesarios para alimentar, configurar y mantener los elementos de un campo de tipo **Lista externa** llamado **`project`**.

- Para consultar los valores que hay disponibles para ese campo:
    - GET **/api/external-lists/project**
- Para consultar los valores que hay disponibles para ese campo, junto con los usuarios asociados a dichos valores:

    - GET **/api/external-lists/project?with=users**

- Para consultar los valores que tiene disponibles un usuario concreto con ID=<uid>
    - GET **/api/external-lists/project?user=<uid>**

- Para añadir un nuevo elemento a la lista:
    - Global: lo verán todos los usuarios:
        - POST **/api/external-lists/project**
        - ```
          {
                "active": true,
                "global": true,
                "label": "Parte 01",
                "value": "PT202021201",
                "field": "project",
                "company_id": {{companyId}}
          }
          ```
    - A un usuario o usuarios particulares:
        - POST **/api/external-lists/project**
        - ```
          {
                "active": true,
                "global": false,
                "label": "Equinox Fase 2",
                "value": "EQPT202020213",
                "field": "project",
                "company_id": {{companyId}},
                "ids_users": [{{idUser1}},{{idUser2}}]
          }
          ```

- Para desactivar un elemento y ya no pueda usarse:
    - PATCH **/api/external-lists/project/{{idValor}}**
    - ```
      {
            "active": false
      }
      ```

- Para vincular un elemento ya existente a un o varios usuarios:
    - POST **/api/external-lists/project/{{idValor}}/actions/attach-users**
    - ```
      {
            "ids_users": [{{idUser1}},{{idUser2}}]
      }
      ```

- Para desvincular un elemento ya existente de un usuario:
    - POST **/api/external-lists/project/{{idValor}}/actions/detach-users**
    - ```
      {
            "ids_users": [{{idUser3}}]
      }
      ```
      ```

2.2. Gestión de grupos de validación (Departamentos)

Para entender correctamente este apartado es necesario tener claros ciertos conceptos manejados en el proceso de validación de un hoja de gastos dentro de Okticket:

1. **Estados**: Situación actual de una hoja de gastos que típicamente tendrá los valores de *Abierta, Pentie de de Revisión, En Revisión, Aprobada, Rechazada y Tramitada.*
2. **Transiciones**: Movimiento o acción de cambiar una hoja de un estado a otro.
3. **Responsable de transición**: Como indica su nombre es el responsable de realizar la transición de un estado a otro. Puede haber varios por cada transición, y pueden ser Roles o Usuarios.
4. **Flujo de aprobación o validación (workflow o WF):** Concepto que engloba los pasos a seguir para llevar a cabo una validación. Se compone de estados, transiciones y responsables de estos mismos.
5. **Grupos de validación (*departamentos*):** Agrupación de usuarios cuyas hojas de gasto tienen los mismos pasos y responsables de validación, es decir, el mismo flujo de validación.

Dado que un **grupo de validación** se utiliza para agrupar a los usuarios cuyas hojas de gasto tienen el mismo flujo de validación, en cada grupo deben estar incluidos tanto los usuarios propietarios de la hoja de gastos a validar, cómo los distintos usuarios validadores que participen en el proceso.

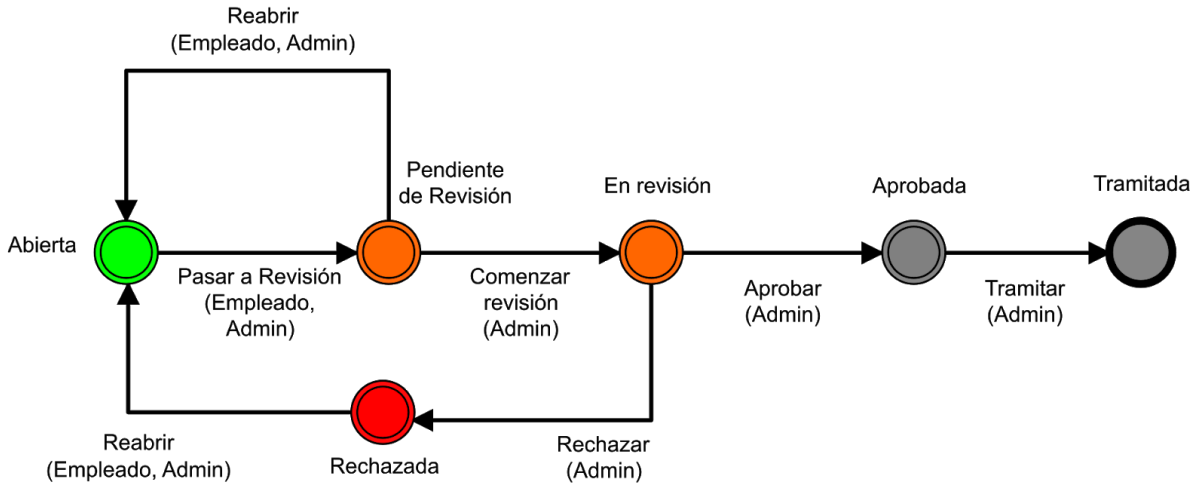
Los pasos y los responsables de las transiciones entre estados quedan definidos por el Workflow que debe estar asociado a cada grupo.

En Okticket hay ciertos WFs predefinidos que pueden usarse, aunque también pueden diseñarse nuevos para adaptarse los procesos de validación de los distintos clientes.

Se muestran a continuación un par de ejemplos basado en los WF por defecto de Okticket:

2.2.1. Workflows

WF por defecto

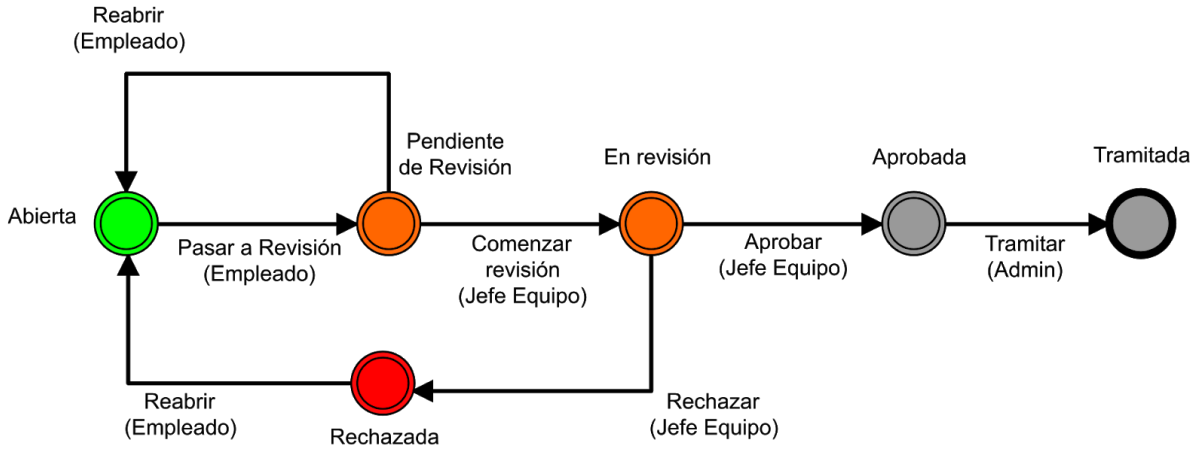


Roles: Empleado (ID: 3) y Administrador (ID: 2)
Uso: Para aquellos clientes en los que solo hay un validador o un solo paso de validación.
Modalidades:

1. Un validador único par toda la empresa: no hace falta crear grupos de validación, todos los usuarios tienen el mismo flujo a nivel de empresa y el validador es siempre el mismo: un administrador a nivel de empresa.
2. Un validador por grupo: Se añaden a cada grupo todos los usuarios como empleados y su validador como Administrador del grupo.

Para este ejemplo, solo aplicaría el segundo caso, ya que es el único en que se usan grupos de validación.

WF con jefes de equipo



Roles: Empleado (3), Jefe de Equipo (6) y Administrador (2).
Uso: Para aquellos clientes en los que solo haya **dos pasos de validación**.
Modalidades:

1. Un primer validador por grupo, y un segundo validador único para toda la empresa: Se añaden a cada grupo todos los usuarios del mismo como Empleados y a su validador como Jefe de equipo. El segundo validador es Administrador a nivel de empresa, no hace falta asociarlo a cada grupo.
2. Dos validadores distintos por grupo: Se añaden a cada grupo todos los usuarios del mismo como Empleados, el primer validador como Jefe de equipo, y el segundo validador como Administrador.

2.2.2. Ejemplos de llamadas a la API

Crear nuevo grupo

- POST **/api/departments**

```
{
      "company_id": <id_empresa>,
      "name": "Grupo proyecto Alfa"
}
```

- Devuelve un JSON con los datos del grupo creados

```
{
    "data": {
        "id": 1295
        "company_id": <id_empresa>,
        "name": "Grupo proyecto Alfa",
        "updated_at": "2020-04-24 11:44:32",
        "created_at": "2020-04-24 11:44:32",

    },
    "status": "ok"
}
```

Añadir empleado a un grupo

Para añadir el usuario <id_usuario> como Empleado del grupo <id_grupo>, hay que realizar la siguiente llamada:

- PATCH **/api/users/<id_usuario>**

```
{
    "ids_departments": {
      "<id_grupo>": {
```

```
            "id_role": 3,
            “web_access”: 1,
            “app_access”: 1
        }
    }
}
```

- El parámetro APP_ACCESS debe ir a 1 para permitir imputar gastos al usuario en el grupo desde la APP.
- Un mismo usuario puede pertenecer a varios grupos de validación, en cuyo caso, deberá elegir grupo al crear los gastos desde la APP.

Añadir administrador a un grupo

Para añadir el usuario <id_usuario> como Administrador del grupo <id_grupo>, hay que realizar la siguiente llamada:

- PATCH **/api/users/<id_usuario>**

```
{
    "ids_departments": {
        “<id_grupo>": {
            "id_role": 2,
            “web_access”: 1,
            “app_access”: 0
        }
    }
}
```

- El parámetro APP_ACCESS debe ir a 0 ya que los validadores no imputan gastos en dicho grupo desde la APP.
- Puede haber más de un administrador por grupo. Cualquiera de ellos podrá realizar las validaciones.

Añadir Jefe de equipo a un grupo

Para añadir el usuario <id_usuario> como Jefe de Equipo del grupo <id_grupo>, hay que realizar la siguiente llamada:

- PATCH **/api/users/<id_usuario>**

```
{
    "ids_departments": {
        “<id_grupo>": {
            "id_role": 6,
            “web_access”: 1,
            “app_access”: 0
        }
    }
}
```

- El parámetro APP_ACCESS debe ir a 0 ya que los validadores no imputan gastos en dicho grupo desde la APP.
- Puede haber más de un jefe de equipo por grupo. Cualquiera de ellos podrá realizar las validaciones.

Obtener los usuarios asignados a un grupo

Para obtener todos los usuarios del grupo <id_grupo>, hay que realizar la siguiente llamada:

- GET **/api/departments/<id_grupo>/users**
- Devuelve un JSON con los usuarios del grupo,

```
{
    "data": [{
        "id": <id_user_1>,
        "id_role”: 3,
        “name”: “Manuel Rodiguez”,
        “email”: “mrodriguez@here.com”
       ...

    },{
        "id": <user_2>,
        "id_role”: 3,
       ...

    },
...
    {
        "id": <user_1>,
        "id_role”: 2,
       ...
    }],
...
}
```

Obtener los grupos a los que está asignado un usuario

Para obtener todos los grupos a los que está asignado el usuario <id_usuario>, hay que realizar la siguiente llamada:

- GET **/api/users/<id_user>/departments**
- Devuelve un JSON con los grupos del usuario:

```
{
    "data": [{
        "id": <id_group_1>,
        "name”: “Group 1”,
```

```
        ...

    },{
        "id": <id_group_2>,
        "name": “Group 2”,
        ...

    },
...
    {
        "id": <id_group_3>,
        "name": “Group 3”,
        ...
    }],
...
}
```

Eliminar usuarios de un grupo

No existe un método específico que permita eliminar al usuario <id_user> del grupo <id_grupo>. Hay que hacer un PUT para actualizar los usuarios del departamento y dejar solo aquellos en que se quiera mantener:

- PUT **/api/departments/<id_grupo>**

```
{
      "ids_users": [8, 2985]
}
```

Si se quieren borrar todos, se haría de la siguiente manera, por tanto:

- PUT **/api/departments**

```
{
      "ids_users": []
}
```